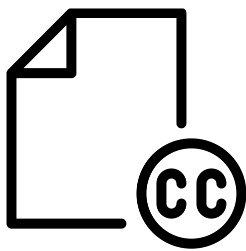


Taller Cooperativisme de Plataforma

Estrategies de Desenvolupament

Autoria: Enrico Stano (CoopDevs),
Pau Pérez (CoopDevs)

Barcelona Activa

CONTINGUTS



Repassem els fonaments de desenvolupament de software, de mètodes de desenvolupament en obert de programari lliure amb equips distribuïts, treballant en comunitats obertes. Toquem breument arquitectures de xarxes distribuïdes i el seu potencial per plataformes col·laboratives.

CONCEPTES CLAU



- Arquitectura d'apps distribuïdes
- Bases de dades
- Apps Mòbils
- Metodologies de desenvolupament
- Operacions i administració de sistemes

Client/Servidor



És un model d'aplicacions centralitzades que particiona una tasca entre dues parts en el que la comunicació entre ambdos parts succeix a través de la xarxa.

Aquest model és la base fonamental de moltes tecnologies que fem servir durant el nostre dia a dia.

Funciona de la manera següent:

- El servidor arrenca un procés de vida indefinida escoltant peticions en un port de xarxa determinat.
- El client envia una petició a aquest mateix port i espera una resposta.
- El servidor processa la petició i retorna una resposta al client.

Frontend

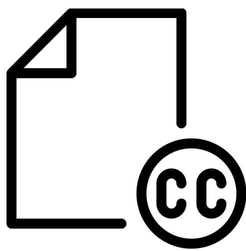
En el cas d'aplicacions web el client rep habitualment el nom de frontend perquè és a la banda del client on s'executa la part visual amb la que interactua l'usuari.

El codi frontend s'executa al navegador i és el que implementa el comportament de la pàgina així com els botons, diàlegs, camps de text, animacions, etc. aquest conjunt d'elements d'interacció conformen el que s'anomena interfícia d'usuari. És senzillament el mitjà on es produeix la comunicació entre usuari i màquina.

Backend

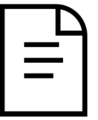
Backend per contra és el nom que rep el servidor en el context d'aplicacions web. Rep aquest nom pel fet que executa la part que no és visual, com en el cas del frontend.

El codi backend s'executa a màquines dedicades exclusivament per aquesta finalitat, molt sovint contractades a tercers, i és el que implementa la lògica de negoci. Tot i que aquest fet no és exclusiu del backend en segons quines arquitectures d'aplicació, sí que és el cas de la web clàssica. Una altra responsabilitat del backend és executar la base de dades.



Autoria: Enrico Stano (CoopDevs),
Pau Pérez (CoopDevs)

Base de dades



Les dades són la peça fonamental de qualsevol aplicació web. Sense les dades no hi pot haver aplicació. En la gran majoria dels casos l'app haurà de persistir. És per això que tant la gestió eficaç de les dades com un disseny adeqüat de la seva estructura són crítics pel bon funcionament de l'aplicació.

Les aplicacions web clàssiques fan servir bases de dades relacionals. Un tipus de bases de dades que fa servir el llenguatge estàndard SQL per mantenir la base de dades i obtenir-ne informació.

Així doncs, SQL permet comunicar l'usuari amb la base de dades a través de quatre tipus d'operacions: SELECT, INSERT, UPDATE, DELETE.

Per altra banda, les bases de dades relacionals també disposen de quatre operacions bàsiques per a la modificació de l'estructura de la base de dades: TRUNCATE, DROP, ALTER i CREATE. Aquestes operacions reben el nom de "llenguatge de definició de dades".

Les bases de dades en general, no només les relacionals, es basen també per la seva banda en el model client/servidor. El sistema gestor de la base de dades s'executa de manera indefinida al servidor esperant peticions en un port especificat. L'usuari utilitza un client compatible amb aquest sistema gestor de base de dades per enviar-hi peticions usant les operacions anteriorment esmentades. És llavors quan el sistema gestor respon a la petició del client amb el resultat de l'operació, i aquest per la seva banda les mostra visualment a l'usuari en forma de taula.

Applicacions natives

En el context de les aplicacions mòbils el concepte "natiu" fa referència a la tecnologia del fabricant: Android, iPhone, etc.

Les tecnologies multiplataforma per contra, fan ús de tecnologies web d'ampli ús i que no depenen dels fabricants per obtenir aplicacions que es poden executar a diverses plataformes amb un sol codi font.

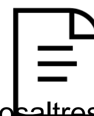
API

De l'anglès "Application Programming Interface" és la implementació del conjunt d'operacions que permet un backend, és a dir, la lògica de negoci.

A diferència de la web clàssica, com ja s'ha vist, no es tracta d'una interfície d'usuari que permet la interacció home-màquina, sinó que només proporciona les dades per a que les processi el client. Serà el client el que després generarà la interfície d'usuari adequada per permetre aquesta interacció.

Exemple https://jsonplaceholder.typicode.com/posts

Reusant codi



És fonamental entendre que no cal que tot el codi que el nostre producte faci servir sigui escrit per nosaltres. La majoria d'aspectes que cal implementar en una aplicació web ja estan resolts de manera molt eficient i les seves implementacions estan publicades sota llicències lliures.

Qualsevol intent de reinventar-ho no tindria sentit ja que no hi afegiria valor i segurament contindria més errors de funcionament i rendiment que sol·lucions ja establertes. És més útil i beneficiós per l'èxit de la nostra plataforma centrar els nostres esforços en aquelles coses que aportin més valor.

Cal dir però, que sovint no s'és conscient del fet que s'està reinventant la roda i sense treball en equip pot ser difícil adonar-se'n.

Git

Per poder reusar codi i fer codi de manera col·laborativa cal emmagatzemar-lo fora del nostre ordinador i fer-lo accessible a la rest de membres de l'equip.

Git permet la gestió de canvis en arxius i directoris al llarg del temps, de manera que podem recuperar l'estat que teni el codi en un moment concret.

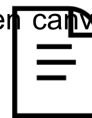
En la nomenclatura de Git, rep el nom de "commit" la fotografia de l'estat actual del codi. Aquestes fotografies, es van acumulant en el que s'anomena "branca". És senzillament un conjunt de commits concret i n'hi pot haver diverses. Per últim, el repositori és el conjunt de branques que conté tot el codi i informacions del nostre projecte de programari.

Metodologies àgils

Com ja s'ha introduït als apartats anteriors, el cooperativisme de plataforma es refereix a projectes que tenen un fort enllaç amb les comunitats en les quals es generen. A l'hora de disenyar un projecte tecnològic es recomana implicar a tots els actors afectats: usuàries, persones amb perfils tècnics i tota la comunitat.

Els processos de co-disseny solen utilitzar les metodologies àgils per incloure les necessitats d'una comunitat en els processos de desenvolupament i traduir-los en iteracions composades per tasques concretes i fórmules per mesurar els resultats.

A cada iteració s'implementa un petit grup de canvis, que aporti valor a la comunitat i que sigui facil d'avaluar. La fase de avaluació de cada iteració alimenta les següents iteracions per incorporar les correccions i adaptacions que hagin sorgit. D'aquesta manera és més facil integrar la comunitat en el co-disseny ja que solsament cal avaluar petit canvis i comparar-los amb la visió global del projecte.



Abordar grans canvis intentant aportar el màxim valor possible en una única iteració molt llarga pot en canvi, resultar molt difícil d'avaluar per un grup de persones nombrós.

Àgil i obert

Les metodologies àgils ja aporten un enorme valor per si soles, motiu pel qual han tingut un gran èxit en el món tecnològic empresarial no limitat a les petites i mitjanes empreses.

La combinació entre aquestes metodologies i el desenvolupament en obert aporten un potencial molt més gran en el món del cooperativisme de plataforma.

Dividir el desenvolupament del producte en tasques d'abast molt reduït i definit, com part d'iteracions englobades en el full de ruta de la plataforma permet tant desenvolupar el producte entre varies persones fins i tot en les mateixes àrees del programari, com un desenvolupament veloç amb molts canvis incrementals i freqüents.

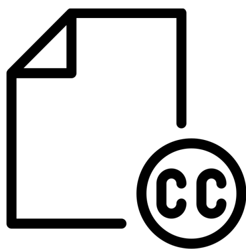
Les tasques petites tenen a més el benefici afegit de no requerir un esforç cognitiu excessiu per a cap de les parts implicades. El desenvolupador entén perfectament quins són els requeriments de la tasca i compleix amb les expectatives sense haver de preguntar a la persona que ha creat la tasca. Per altra banda, la persona que s'encarrega de provar la implementació entén ràpidament quines són les parts afectades i quines funcionalitats s'han de provar.

No menys important és l'impacte positiu que té a l'estat d'ànim dels membres de l'equip de desenvolupament i la comunitat en general veure com el producte evoluciona positivament i s'avança en els objectius marcats, i com el treball en equip és efectiu i àgil.

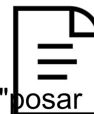
Per últim, tasques d'abast reduït i ben definit són imprescindibles per tal d'obrir la porta a contribucions per part de membres de la comunitat com també de contribucions esporàdiques per part de gent fora de la comunitat.

Un cop fet el triatge de tasques fàcils per nous contribuïdors cal encoratjar a la gent a fer el pas de participar del desenvolupament, no només en codi. Participar de manera activa requereix reconèixer les pròpies mancances, deslligar-se de les pròpies vergonyes i centrar-se en les potencialitats que es poden aportar al projecte.

Quan això va seguit d'un acompanyament augmenten molt les probabilitats que de ser un contribuidor esporàdic es passi a ser un membre més de la comunitat. Cal tenir en compte però, l'esforç i dedicació que tot això requereix dels membres més actius i de l'equip de desenvolupament.



Operacions i administració de sistemes



Un dels aspectes menys evidents en tot el procés de producció de programari està relacionat amb "posar a producció" el mateix sofwate. És a dir, posar-lo a disposició dels usuaris. En funció de l'arquitectura de xarxa i de l'aplicació que s'hagi escollit hi ha diverses maneres de portar a terme aquest procés.

En el de client/servidor, l'equip responsable d'entregar el producte als seus usuaris necessita traslldar el programari a servidors accessibles des de internet. Els processos necessaris per fer-ho estan estrament relacionats amb el conjunt de tecnologies escollides per donar suport al programari produït.

Tot i aquesta espcificitat, hi ha una sèria de patrons recurrents en aquests processos. És important però, adoptar unes pràctiques que permetin a tot l'equip conèixer aquests processos de manera que no depenguin de persones concretes.

Els processos bàsics necessaris per posar un programari a producció es poden resumir en:

* Gestió de servidors
* Provisionament
* Desplegament
* Monitorització

Gestió de servidors

L'elecció dels servidors en els que s'executarà el programari requereix coneixement del mercat i dels factors relacionats amb el posicionament geogràfic dels usuaris del programari. Entre tants proveïdors de serveis d'allotjament es necessari entendre també el nivell de confiança que podem esperar segons el model de governança, la mida de l'empresa, etc.

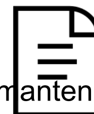
Provisioning

Una vegada aconseguit accés als servidors, necessitem preparar les màquines perquè puguin rebre i executar el nostre programari. El procés de provisionament s'encarrega d'instal·lar en els servidors tot aquell programari que no hem desenvolupat nosaltres directament però que necessitem per executar el nostre. Un exemple pot ser el programari que gestiona les bases de dades, etc. Aquest programari se sol anomenar dependència.

Desplegament

A partir del moment en què els servidors estan equipats amb totes les dependències necessàries podem començar el procés de trasllat del programari produït pel nostre equip. Aquest procés és responsable de posar físicament a producció el programari. Un aspecte important d'aquest procés és que ha de facilitar l'accés a la versió del programari que ha estat desplegada i a processos d'emergència per tornar a activar versions anteriors.

Monitorització



Hauria de ser ja bastant evident que tot el conjunt de programari i processos necessaris per mantenir accessible el nostre programari pot resultar difícil de mantenir i sobretot difícil de gestionar en casos d'emergències.

Per reduir aquesta amenaça se solen engegar mecanismes i processos de monitorització. Es tracta d'instrumentar i monitoritzar quants mes aspectes possible, partint del correcte funcionament del servidor (memòria, CPU, etc) fins a cada peça del nostre conjunt de programari que resulta vital per al nostre projecte.

Obtenint totes aquestes dades es pot engegar un sistema d'alertes que avisi a l'equip cada cop que alguns valors mesurats s'allunyen dels valors establerts. Aquestes dades i informacions també són importants a l'hora d'analitzar un problema ocorregut i per actuar perquè no es torni a repetir.

Processos compartits

Acabem d'entendre la complexitat i alhora pes fonamental que tenen els processos de posada a producció d'un programari, una fallada en aquest conjunt pot posar en risc tot el treball invertit en el desenvolupament del nostre projecte. Per aquesta raó es recomanable adoptar bones pràctiques i compartir el coneixement entre projectes que usen tecnologies semblants.

Documentar-ho tot

Centralitzar el coneixement pot posar en perill un projecte tecnològic d'aquest tipus. Cal contrarestar qualsevol tipus de resistència a compartir la informació encara que pugui resultar en una perduda de poder per a algunes persones. És important documentar cada decisió i cada canvi en cadascun dels processos esmentats. Si per alguna raó hi ha parts d'aquests processos que es van implementar sense generar documentació cal aprofitar l'ocasió per demanar a una persona que mai hagi tingut accés a aquests processos de documentar-los. La millor persona per documentar un procés és la que mai els hagi posat en pràctica, és una bona ocasió per transmetre coneixement i alhora generar documentació clara i entenedora.

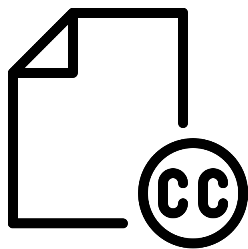
Reusar processos entre projectes

En els casos en els quals aquesta cultura de compartir coneixement no hagi estat adoptada per tot l'equip, ens podem trobar amb processos que solucionen problemes de forma ineficient. Generalment és a causa de persones que desconeixien o ignoraven l'existència de processos semblants i la falta de comunicació. És important reutilitzar els processos entre els projectes i trobar la forma d'implicar totes les persones interessades quan es treballa en la implementació de nous processos.

Fàcil replicació

Des d'una perspectiva del pro-comú digital tot aquest treball i esforç dedicat a documentar i compartir els processos de producció és necessari que es faci visible també fora de la pròpia organització publicant tot el que es pugui amb llicències obertes (Creative Commons, etc). D'aquesta forma es podrà potenciar la replicació dels projectes ja que, com hem vist, no solament el programari en si és important i estratègic sinó que també ho són tot el conjunt de processos per a la posada a producció i el manteniment del projecte. Exemple: https://github.com/loomio/loomio-coop-handbook





Autoria: Enrico Stano (CoopDevs),
Pau Pérez (CoopDevs)

Barcelona Activa

RECURSOS ÚTILS			
Títol i accés	**Utilitat?** ⋆	⋆⋆	⋆⋆⋆
Building on Blockchain (Resonate) https://resonate.is/building-on-blockchain/			
Intro to Crypto-economics (Peter Harris) https://medium.com/@peteratomic/intro-to-crypto-economics-9508e471d617			
Loomio Cooperative Handbook https://github.com/loomio/loomio-coop-handbook			
What is co-design (Design for Europe) http://designforeurope.eu/what-co-design			
Design Sprint Kit (Google) https://designsprintkit.withgoogle.com/			
Triage issues in 7 simple steps (GitLab) https://about.gitlab.com/2017/10/26/triage-issues-gitmate/			
Using Loomio to govern a self-organising community (Enspiral) https://medium.com/enspiral-tales/using-loomio-to-govern-a-self-organising-community-45ef4af15f26			
Blockchain Doesn't Decentralise Power… Unless You Design It To (Enspiral) https://medium.com/enspiral-tales/blockchain-doesnt-decentralise-power-5918c168e6f6			